Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 21 (2024) 274-285

www.revista-riai.org

Arquitectura Software para el Sistema Robótico de Manipulación Dual HortiRobot

Daniel Rodríguez-Nieto[a,b,*], Marta Ojeda[a], Eduardo Navas[a], Roemi Fernández[a]

[a] *Centro de Automática y Robótica CAR CSIC-UPM, Ctra. Campo Real Km 0,200 La Poveda, Arganda del Rey, 28500, Madrid, España*
[b] *Universidad Politécnica de Madrid, Programa de Doctorado en Automática y Robótica, Calle de José Gutiérrez Abascal 2, 28006, Madrid, España*

To cite this article: Rodríguez-Nieto, D., Ojeda, M., Navas, E., Fernández R. 2024. Software Architecture for the HortiRobot Dual-Arm Robotic Manipulation System. Revista Iberoamericana de Automática e Informática Industrial 21, 274-285. https://doi.org/10.4995/riai.2024.20611

Resumen

La arquitectura software es un componente crucial en cualquier sistema robótico autónomo, ya que define la estructura organizativa y las interacciones de los diferentes módulos que lo integran. Para que un sistema robótico pueda ejecutar de forma autónoma diversas tareas, se requieren procesos variados, como percibir el entorno, representar conocimientos, tomar decisiones y planificar movimientos. Si bien el desarrollo de cada uno de estos procesos es fundamental, su integración en una arquitectura funcional para su implementación también lo es. Esta integración tiene profundas implicaciones en la gestión de recursos, la adaptabilidad a diferentes entornos y tareas, la flexibilidad para modificar o expandir las funcionalidades y hacer frente a nuevos requerimientos, y la facilidad para el mantenimiento y la actualización del sistema. Por ello, en este artículo se presenta la arquitectura software diseñada para controlar, comunicar e integrar los distintos módulos que componen un bimanipulador móvil, destacando entre sus principales ventajas, la facilidad para depurar errores y llevar a cabo pruebas de nuevas aplicaciones sin el riesgo inherente de dañar el equipo físico. Para demostrar la viabilidad de la propuesta, la implementación de la arquitectura se valida mediante su aplicación al sistema robótico de manipulación dual HortiRobot, concebido para realizar varias de las tareas implicadas en el ciclo de vida de los cultivos agrícolas.

Palabras clave: Robótica agrícola, arquitectura software, manipulación robótica dual, percepción inteligente, inteligencia artificial.

Software Architecture for the HortiRobot Dual-Arm Robotic Manipulation System.

Abstract

The software architecture is a crucial component in any autonomous robotic system, as it defines the organizational structure and interactions of the different modules that compose it. For an autonomous robotic system to perform various tasks, diverse processes are required, such as perceiving the environment, representing knowledge, making decisions, and planning movements. While the development of each of these processes is fundamental, their integration into a functional architecture for implementation is equally important. This integration has profound implications for resource management, adaptability to different environments and tasks, flexibility to modify or expand functionalities and address new requirements, and ease of system maintenance and updates. Therefore, this article presents the software architecture designed to control, communicate, and integrate the various modules that make up a mobile bimanipulator, highlighting among its main advantages the ease of debugging errors and conducting tests for new applications without the inherent risk of damaging the physical equipment. To demonstrate the feasibility of the proposal, the implementation of the architecture is validated through its application to the mobile dual-arm robotic system HortiRobot, designed to perform various tasks involved in the life cycle of agricultural crops.

Keywords: Agricultural robotics, software architecture, dual-arm robotic manipulation, intelligent perception, artificial intelligence, learning.

*Autor para correspondencia: daniel.r.nieto@csic.es

1. Introducción

La arquitectura software define cómo se organizan e interactúan entre sí los diferentes módulos, componentes y subsistemas de software para cumplir con las funcionalidades definidas (Richards, 2015). Si hablamos de un sistema robótico, la arquitectura software proporciona un marco para organizar y coordinar las diversas partes del software que permiten que un robot realice tareas específicas (Ferrati et al., 2016). En el caso concreto de los robots manipuladores móviles, se obtiene una mayor flexibilidad, al permitir que la manipulación robótica sea independiente de la ubicación. Sin embargo, esta flexibilidad conlleva un aumento significativo de la complejidad, ya que se necesita gestionar simultáneamente varios sensores y actuadores para ejecutar las tareas planificadas y responder a cambios inesperados en el entorno. Para ejecutar todos estos procesos de forma simultánea y asíncrona, se requiere de una arquitectura bien concebida que ayude a gestionar esta complejidad, proporcionando modularidad, reconfigurabilidad, y fiabilidad (Hendrich et al., 2015). Además, contar con un conocimiento profundo de la arquitectura facilita la identificación de fallos y la corrección de errores, y simplifica el proceso de mantenimiento, asegurando que el robot funcione de manera continua sin errores.

Una arquitectura correctamente definida permite a los sistemas controlar diversos actuadores, interpretar la información adquirida por los sensores, planificar acciones y movimientos, monitorizar la ejecución y manejar contingencias y eventos inesperados. Además, proporciona el marco conceptual para el desarrollo de software dependiente del dominio y a menudo ofrece herramientas de programación que facilitan ese desarrollo (Kortenkamp et al., 2016).

Otro aspecto relevante es la optimización de recursos. Entender cómo se estructura el software del robot permite identificar posibles cuellos de botella o ineficiencias en el sistema. Esto es esencial para aprovechar al máximo los recursos disponibles, como la capacidad de procesamiento, la memoria y la energía, lo que puede llevar a un mejor funcionamiento (Cervantes et al., 2016). Además, la arquitectura de software proporciona una base sólida para el desarrollo de nuevas funcionalidades. Si se planea agregar capacidades adicionales al robot o desarrollar aplicaciones específicas, comprender cómo integrar estas funcionalidades en la arquitectura existente es constitutivo. Esto facilita la expansión y adaptación del sistema a medida que cambian los requisitos y las necesidades.

La seguridad es otro aspecto crítico en la robótica, especialmente cuando los robots interactúan con entornos complejos y cambiantes. Comprender la arquitectura de software es fundamental para implementar medidas de seguridad adecuadas y garantizar que el robot pueda funcionar de manera segura en diversas situaciones, minimizando riesgos potenciales (Gribov y Voos, 2014).

Asimismo, es común que los sistemas robóticos interactúen con otros dispositivos y sistemas. La comprensión de la arquitectura de software es indispensable para asegurar la interoperabilidad y la comunicación efectiva con otros componentes y sistemas (Nakagawa et al., 2021).

Aunque actualmente no existe una arquitectura única de referencia para robots manipuladores móviles, una característica común que presentan es la descomposición modular en partes más simples e independientes. Este diseño permite que los procesos presentes en los diferentes módulos manejen las interacciones con el entorno de forma asíncrona, al tiempo que minimizan las interacciones entre sí. Por lo general, esto aumenta la fiabilidad global. A menudo, la descomposición del sistema es jerárquica, es decir, los componentes modulares se construyen sobre otros componentes modulares. Las arquitecturas que admiten explícitamente este tipo de descomposición en capas reducen la complejidad del sistema a través de la abstracción (Pramod et al., 2015), (Medvidovic et al., 2011).

Este uso de la división por capas para el diseño de una arquitectura de software puede verse en algunos de los últimos trabajos realizados en manipuladores duales. Por ejemplo, en el trabajo de Aitor Ibarguren (Ibarguren et al., 2020) se presenta una arquitectura de tres capas para un robot colaborativo manipulador dual con comunicación humano-robot, que consta de: (i) una capa de control de guía, encargada del control de bajo nivel del robot; (ii) una capa de gestión de la información, que recolecta todos los datos en tiempo real y genera información significativa para ser utilizada como realimentación; y (iii) una capa de interfaz de usuario. Existen algunos desafíos técnicos inherentes a esta propuesta, como la dificultad de coordinación de ambos brazos robóticos, y el correcto intercambio de información entre humanos y robots.

Otra arquitectura similar se presenta en el robot móvil con manipulador dual para tareas de asistencia de Gillini (Gillini et al., 2022). En este caso, el enfoque seguido es el de emplear ROS como middleware, comunicando el lector BCI (*Brain Computer Interface*) con el robot manipulador dual y el sistema de visión artificial. La arquitectura se compone de los módulos de percepción, manipulación y control neuronal. La estrategia seguida es capaz de gestionar tareas basadas en conjuntos, así como tareas organizadas de manera jerárquica.

Un trabajo más reciente es el desarrollado por Jianguo Duan, (Duan et al., 2023), en el que se crea un marco para monitorizar un robot colaborativo dual, mediante el uso de un gemelo digital. En este caso, también se trabaja con un sistema por capas, basado en el modelo de capas de comunicación OSI con algunas modificaciones. Las capas son las siguientes:

- Capa física: involucra el espacio físico con entidades de montaje y equipos auxiliares.
- Capa virtual: se encarga de la construcción de modelos 3D y de la interacción en un espacio virtual.
- Capa de comunicación: facilita la transmisión de datos mediante redes, protocolos de comunicación y dispositivos de entrada/salida.
- Capa de datos: incluye atributos físicos y datos virtuales, junto con la información del robot. Ofrece datos derivados a través de simulaciones.
- Capa de aplicación: proporciona servicios funcionales y empresariales.
- Capa de usuario: permite la interacción humano-robot.

En este trabajo se demostró la viabilidad del sistema para lograr la monitorización en tiempo real del proceso de

ensamblaje, y su utilidad como apoyo para la toma de decisiones inteligente.

Otro ejemplo destacado de un enfoque innovador se presenta en el trabajo reciente de Serdar Kalaycioglu y Anton de Ruiter (Kalaycioglu y Ruiter, 2023). En este estudio se emplean brazos robóticos como manipuladores espaciales redundantes instalados en una nave espacial. La arquitectura propuesta abarca: (i) un Sistema de Determinación y Control de Actitud de la Nave Espacial (ADACS) junto con un Sistema de Control del Robot (RCS), (ii) un sistema de Manejo de Comandos y Datos (C&DH), y (iii) un procesador de comunicaciones. Esta estructura modular tiene la ventaja de reducir la complejidad del sistema, permitiendo así realizar un análisis más detallado en cada uno de los componentes. No obstante, coordinar manipuladores duales para ensamblajes en órbita presenta grandes complicaciones respecto al cálculo de fuerzas y pares en los actuadores. Por ello, se requiere el empleo de algoritmos que optimicen la coordinación.

Como se puede apreciar tras lo comentado en esta sección, la definición de las arquitecturas robóticas es muy variable, y depende en gran medida del objetivo para el que está destinado el robot empleado. Sin embargo, sí que se observa una tendencia general a adoptar sistemas modulares jerarquizados, empleando distintas capas programadas individualmente que serán interconectadas para comunicar todos los módulos del robot, habilitando así su funcionalidad completa.

En este artículo se presenta la arquitectura software diseñada para controlar, comunicar e integrar los distintos módulos que componen el sistema robótico de manipulación dual propuesto en HortiRobot, financiado por la Agencia Estatal de Investigación, y concebido para realizar varias de las tareas implicadas en el ciclo de vida de los cultivos agrícolas.

La agricultura ha sido una actividad de vital importancia en la historia de la humanidad, desempeñando un papel fundamental en el desarrollo y sustento de las sociedades al proporcionar alimento para millones de personas en todo el mundo. España destaca como el principal exportador de productos hortofrutícolas en la Unión Europea, y se sitúa entre los tres principales exportadores a nivel global, junto a China y Estados Unidos. El valor de la producción en el sector de frutas y hortalizas, excluyendo uva para vinificación y aceitunas, experimentó un incremento significativo en 2021, superando los 15.400 millones de euros. Este dato representa un aumento del 6% con respecto a 2020 y un incremento del 9% en comparación con la media de los años 2016-2020 (ver Figura 1). Por otro lado, la superficie destinada al cultivo de frutas y hortalizas en España alcanzó las 1.873.520 hectáreas en el año 2021, reflejando un incremento del 3% en relación con el año anterior y un crecimiento cercano al 7% en comparación con el promedio de los últimos cinco años. (Eurostat, 2021), (Ministerio de Agricultura, Pesca y Alimentación, 2021), (Sadjadi y Fernández, 2023).

No obstante, el sector agrícola se ve amenazado continuamente por la baja rentabilidad y la escasez de mano de obra. La crisis sanitaria originada por la COVID-19 ha exacerbado aún más esta situación, resaltando la urgente necesidad de buscar soluciones tecnológicas que permitan automatizar las tareas agrícolas. En respuesta a este desafío, se han realizado considerables esfuerzos en investigación e innovación en los últimos años con el objetivo de introducir sistemas robóticos en el sector agroalimentario. La finalidad de estos esfuerzos es compensar la mencionada escasez de mano de obra necesaria para el cuidado de los cultivos. A pesar de los avances alcanzados en este campo, la adopción generalizada de la mayoría de los enfoques propuestos ha sido limitada, debido a los altos costes y la falta de flexibilidad de los sistemas diseñados. Para superar estas barreras, las soluciones robóticas deben ser capaces de adaptarse a una amplia variedad de actividades, entornos y tipos de cultivos. La respuesta a estas demandas se encuentra en un diseño modular y reconfigurable que permita mantener los costes relativamente bajos, aplicando una configuración básica a una amplia gama de tareas agrícolas.

Por tanto, el objetivo global de la propuesta HortiRobot es incrementar el rendimiento y la sostenibilidad de los cultivos y contribuir a la adopción generalizada de tecnologías robóticas en el sector agroalimentario mediante el despliegue de un sistema robótico móvil, autónomo y con manipulación dual, capaz de realizar varias de las tareas implicadas en el ciclo de vida de los cultivos.

Revisando el estado del arte de arquitecturas de software para robots móviles que realicen tareas agrícolas, se puede apreciar un mayor interés por este tema en los últimos años. Un ejemplo, es el trabajo presentado en (Phanomchoeng et al., 2020), en el que la arquitectura permite al usuario controlar el robot agrícola desde una interfaz a través de un servicio en la nube. No obstante, la complejidad del robot y de las tareas propuestas no es comparable con la de HortiRobot, por lo que la arquitectura resultante es bastante simple y poco escalable. Otro enfoque, presentado en (Vrochidou et al., 2021),

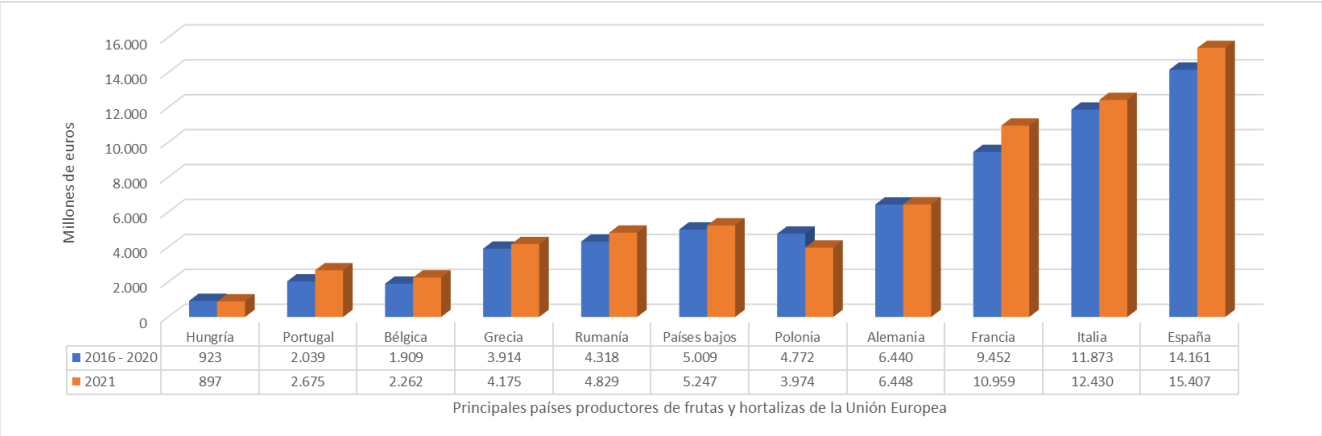

Figura 1: Evolución de la producción de frutas y hortalizas en la Unión Europea en 2021 con respecto al periodo 2016-2020 (Eurostat, 2021).

involucra la gestión de una unidad aérea, una unidad de control remoto y un robot manipulador móvil para la cosecha de uvas. La unidad aérea se encarga de la adquisición de los datos, y con esta información se generan los mapas del viñedo. La unidad de control permite que el usuario planifique la tarea y las rutas con los mapas obtenidos. Finalmente, el robot manipulador móvil ejecuta el plan trazado. La arquitectura de software del robot móvil consta de tres nodos ROS: un nodo para el control del brazo, un nodo para el control de la plataforma móvil y un nodo del subsistema, que vincula el brazo robótico y el efector final y se comunica con la plataforma móvil. Sin embargo, no se describe con suficiente nivel de detalle cómo se estructura la arquitectura a nivel de sistemas: unidad aérea, robot móvil y unidad de control. Además, la solución propuesta tiene la limitación de depender del operador humano para que se haga cargo de la unidad de control remoto, seleccionando las rutas de navegación y las operaciones agrícolas específicas. Por otra parte, en el estudio de Chikurtev (Chikurtev, 2022), se propone una arquitectura de software distribuido para controlar un robot modular para aplicaciones agrícolas y para cría de animales. La comunicación está basada en servicios y microservicios y respuestas entre los módulos individuales. No obstante, el sistema sólo cuenta con tres módulos, un módulo de visión, un módulo de control y un módulo de navegación. Por tanto, la arquitectura resultante también es mucho más simple que la propuesta para HortiRobot. Otro ejemplo es la arquitectura presentada por Xu (Xu y Li, 2022), que consta de tres módulos implementados en ROS para realizar tareas autónomas básicas: un módulo de control, un módulo de navegación y un módulo de visión. Esta arquitectura está bien definida y tiene un concepto similar al propuesto en este artículo, pero su estructura está mucho más simplificada por los requerimientos de las tareas para las que ha sido concebido el sistema robótico MARS. Cabe también destacar el trabajo presentado en (Geer et al. 2022). En este caso, los autores proponen una arquitectura con múltiples componentes independientes, incluyendo el sistema de percepción, la interfaz de usuario, navegación y control del brazo robótico, control del brazo UR3 y navegación. Sin embargo, no se proporciona una descripción clara de cómo están estructurados estos componentes, ni de cómo interactúan entre sí.

Por tanto, teniendo en cuenta la complejidad del sistema propuesto en HortiRobot, es importante contar con una arquitectura de software que permita la composición flexible de diferentes capacidades para llevar a cabo las tareas previstas. Por ello, la arquitectura propuesta estará basada en el modelo de capas, y será concebida para facilitar la implementación, prueba, depuración y evaluación de nuevos algoritmos de percepción, toma de decisiones, manipulación y control en entornos no estructurados, minimizando además el riesgo inherente de daño a los equipos físicos, gracias a un entorno de simulación programado. El concepto de modularidad de la arquitectura, incluido en la etapa de diseño, también contribuirá significativamente a mejorar la fiabilidad y la capacidad de evolución del sistema robótico HortiRobot.

El resto del artículo está organizado de la siguiente manera. En la sección 2 se proporciona una descripción general del sistema HortiRobot. En la sección 3 se explica el diseño de la arquitectura de software propuesta. Finalmente, en la sección 4 se presentan las principales conclusiones y los trabajos futuros previstos.

2. Descripción del sistema

Para llevar a cabo con éxito varias de las tareas implicadas en el ciclo de vida de los cultivos, como el injerto, la poda y la cosecha, el sistema robótico diseñado debe ser capaz de:

- reconocer y localizar en el espacio las plantas y todos sus elementos, en presencia de oclusiones, en condiciones de iluminación variable, y en diferentes etapas del crecimiento;
- localizar las zonas de corte apropiadas para el injerto, la poda y la cosecha;
- detectar las distintas herramientas requeridas para llevar a cabo las tareas propuestas;
- sujetar y retirar la fruta o el elemento requerido de la planta con velocidades comparables a las obtenidas manualmente, pero sin causar daños en el producto o en el cultivo;
- adaptarse a la variabilidad de los productos (tamaño, forma, madurez);
- adaptarse a los cambios del entorno.

Para responder a este reto, la solución propuesta por HortiRobot combina (ver Figura 2):

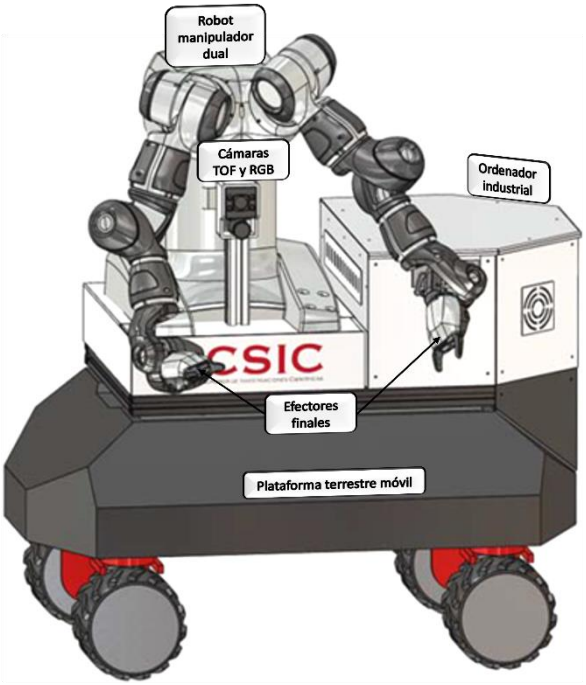

Figura 2: Concepto del sistema HortiRobot.

- Una unidad robótica comercial de doble brazo, compuesta por un torso humanoide y dos brazos de 7 grados de libertad. Dado que los escenarios agrícolas son complejos y dinámicos, interactuar de forma fiable en estos ambientes es difícil. Además, muchas tareas agrícolas requieren el uso de dos brazos y/o dos manos, por lo que es crucial que los robots puedan replicar de manera natural los movimientos humanos en la manipulación. Por estas razones, se ha optado en este caso por un robot

colaborativo YuMi – IRB 14000 de ABB, cuyos brazos tienen un alcance de 559 mm, una capacidad de carga de 500g y una repetitividad de 0,02 mm. Estas características hacen que sea una opción idónea para abordar las complejas demandas de las tareas agrícolas, permitiendo una interacción más natural del robot con el entorno de cultivo.

Figura 3: Sistema de coordenadas de las cámaras y el robot.

– Un sistema de percepción inteligente diseñado para guiar el proceso de toma de decisiones y la interacción del sistema robótico con el entorno. El sistema consta de: (i) una cámara de color Triton, modelo TRI016S de 1,6 MP, con un sensor Sony IMX392 CMOS de 1/2,9", una velocidad de captura de 71,6 fps (fotogramas por segundo), una resolución de 1440 x 1080 píxeles y obturador global para obtener imágenes de alta calidad en tiempo real; y (ii) una cámara de tiempo de vuelo Helios2, que cuenta con un sensor Sony DepthSense IMX556PLR CMOS de 8 mm y un tamaño de píxel de 10,0 μm (H) x 10,0 μm (V), con obturador global. Esta cámara proporciona imágenes en escala de grises con una resolución de 640 x 480 píxeles, una precisión de punto flotante de 16 bits y una velocidad máxima de 30 fps. Además de la imagen en escala de grises, que se corresponde con la respuesta de amplitud, la cámara proporciona un mapa de profundidad del que se pueden extraer las coordenadas Cartesianas XYZ, con el origen del sistema en el centro de la parte frontal de la cámara, con la coordenada Z incrementándose a lo largo del eje óptico a medida que nos alejamos de la cámara, con la coordenada Y incrementándose verticalmente hacia abajo y la coordenada X incrementándose horizontalmente hacia la derecha del robot. En la Figura 3 se puede observar una imagen del robot, con el sistema de referencia de la cámara TOF indicado en la misma. El rango de detección de este dispositivo (distancias radiales) va desde los 0,3 m hasta los 8,3 m, y su campo de visión es de 69° en horizontal y 51° en vertical. Las cámaras, tal y como muestra la Figura 3, están alineadas verticalmente y colocadas lo más cerca posible la una de la otra. Para garantizar la precisión de las mediciones, se han realizado rigurosas calibraciones intrínsecas y extrínsecas para ambas cámaras, siguiendo el método propuesto por Bouguet (Bouguet, 2010). Además, se ha llevado a cabo una calibración de distancias para la cámara de tiempo de vuelo, utilizando el método presentado en el trabajo de Chiabrando (Chiabrando *et al.*, 2009).

– Un conjunto de herramientas y efectores finales modulares, diseñados para realizar las diferentes tareas de manipulación seleccionadas (agarre, corte, desprendimiento). Estas herramientas pueden observarse en la Figura 4. A continuación, se describen brevemente las principales características que las definen:

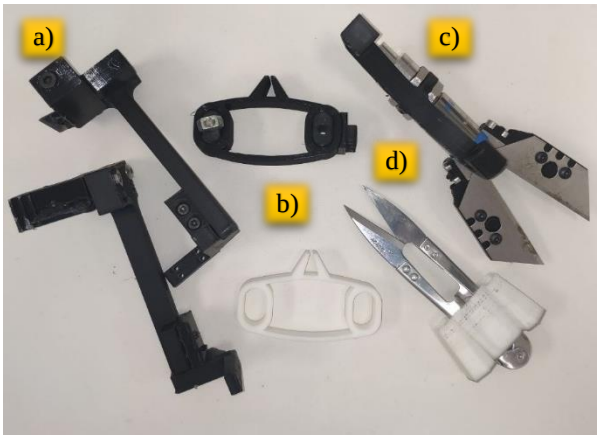

Figura 4: Distintos efectores finales.

a) Efector final para corte y agarre: este efector final ha sido especialmente diseñado para realizar las tareas de injerto. Está compuesto por dos piezas fabricadas en ácido poliláctico (PLA) mediante impresión 3D, las cuales se acoplan al robot YuMi como un reemplazo de sus pinzas originales. Además, cada una de estas piezas consta de dos partes: una parte trasera para el corte del tallo y una parte frontal para el agarre de la pinza de injerto. La parte trasera consta de dos cuchillas alineadas a un ángulo de 45 grados con respecto a la horizontal, colocadas en la cara interna de las piezas. Esta disposición facilita la ejecución del corte del tallo con la inclinación deseada. Una vez realizado el corte, la parte frontal del efector final permite sujetar la pinza del injerto, evitando el contacto con las cuchillas. Además, la geometría del efector garantiza que los brazos izquierdo y derecho puedan operar sin colisionar entre sí.

b) Pinzas blandas: estas pinzas, fabricadas en plástico flexible mediante impresión 3D, han sido concebidas como herramientas para la manipulación de plantas con tallos que tienen un diámetro muy pequeño, y no como efectores finales. Por tanto, para poder utilizarlas, el robot YuMi de ABB debe introducir los dedos de su pinza original (el *SmartGripper*) en cada uno de los orificios elípticos disponibles. De esta forma, cuando se abren los dedos de la pinza original, la pinza blanda se cierra, lo que permite sujetar el tallo. Por el contrario, cuando se cierran los dedos de la pinza original, la pinza blanda se abre, lo que permite soltar el tallo.

c) Tijera neumática: se utiliza específicamente para cortar plantas con tallos leñosos, ya que puede aplicar una mayor cantidad de fuerza en el proceso de corte. Esta tijera se integra al sistema neumático del *SmartGripper* y se activa mediante un compresor de aire. Los adaptadores neumáticos necesarios para su funcionamiento se encuentran ubicados en los laterales del efector final. Una de las ventajas principales de esta herramienta, es que no es necesario inhabilitar los dedos originales del YuMi para su funcionamiento.

d) Accesorio de tijera: se acopla fácilmente al *SmartGripper* mediante un sencillo proceso de montaje y desmontaje automático. Aunque al utilizarlas se elimina por completo la capacidad de sujetar objetos con las pinzas, su facilidad de conexión representa una gran ventaja. Estas tijeras cumplen de manera eficiente con su función de corte.

- Un robot móvil terrestre comercial, que será capaz de llevar a bordo la unidad de doble brazo, el sistema de percepción y el conjunto de efectores finales. El robot móvil seleccionado para HortiRobot es el RB-VOGUI de Robotnik, que tiene motores de tracción en las cuatro ruedas y una autonomía con baterías de hasta 8 horas, lo que garantiza una operatividad prolongada. El robot cuenta con un ordenador a bordo con Linux integrado, que se comunica con otro ordenador externo que hace de maestro. Este ordenador maestro se encuentra alojado en un compartimiento adicional, diseñado específicamente para este propósito, y que no viene incluido de serie, como se puede observar en las Figuras 2 y 3. Ambos ordenadores se comunican vía Ethernet, y trabajan conjuntamente en un entorno ROS, que permite un control eficiente y coordinado de todas las operaciones. El RB-VOGUI es capaz de alcanzar una velocidad máxima de 2,5 m/s, y su capacidad de carga es de 150 kg, por lo que puede transportar de manera efectiva todos los componentes necesarios para llevar a cabo las tareas agrícolas. Estas características hacen del RB-VOGUI un elemento indispensable en el sistema HortiRobot.
- Una arquitectura software para controlar, comunicar e integrar los distintos módulos que componen el sistema robótico HortiRobot. Esta arquitectura permitirá la composición flexible de diferentes capacidades para llevar a cabo las tareas previstas. De esta forma, se simplificará el proceso de creación de comportamientos complejos y robustos del robot en una amplia variedad de tareas y cultivos. El enfoque propuesto facilitará la depuración y realización de pruebas de las nuevas aplicaciones sin riesgo de daño al equipo real. El sistema de control estará basado en el comportamiento coordinado, la comprensión visual del escenario y el aprendizaje, proporcionando una solución versátil para las demandas de la agricultura automatizada. La arquitectura software también incluye una interfaz de usuario para la planificación de las tareas globales y el seguimiento y monitorización de todas las operaciones que realice el sistema robótico.

A continuación, se presenta la arquitectura propuesta para HortiRobot.

3. Arquitectura software

La arquitectura software propuesta para el sistema HortiRobot se compone de tres capas principales: dos de ellas corresponden a diferentes niveles de operación del robot (alto nivel y bajo nivel), mientras que la otra capa está destinada a la interfaz de usuario.

La capa de alto nivel se encarga de gestionar los módulos de procesamiento y algoritmos inteligentes que operan el sistema. Además, proporciona una abstracción del hardware mediante la encapsulación de los controladores ROS adecuados. Esto permite la conexión de los algoritmos de alto nivel con los sensores y actuadores, facilitando el control del sistema robótico mediante el intercambio de mensajes entre procesos.

Por otro lado, la capa de bajo nivel engloba los componentes que establecen una comunicación directa con el hardware del sistema, incluyendo los brazos, sensores, efectores finales y el robot móvil terrestre. Estos elementos son fundamentales para la ejecución de las tareas físicas requeridas en el entorno agrícola.

Para esta implementación, se ha optado por utilizar ROS (*Robot Operating System)*, un *framework* de código abierto que opera en diversas plataformas y cumple una función análoga a la de un sistema operativo en un conjunto de hardware heterogéneo. En Este trabajo, se ha empleado la versión Noetic de ROS, en Ubuntu 20.04.

ROS se basa en una arquitectura de grafos en la que el procesamiento se distribuye en nodos. Cada nodo puede recibir, enviar y multiplexar mensajes provenientes de sensores, el sistema de control, planificaciones y otros componentes relevantes para el funcionamiento del sistema.

En el núcleo de esta arquitectura se encuentra el ROS *master*, que se ejecuta en *localhost* y permite que los nodos se comuniquen entre sí, intercambiando datos de interés. Cada nodo opera con sus propios *topics*, que pueden utilizarse para publicar o suscribir mensajes. Un nodo publica datos en un espacio compartido a través de un *topic*, y los demás nodos pueden suscribirse a ese *topic* para acceder a los datos simultáneamente. Esta estructura de comunicación facilita la coordinación y colaboración entre los distintos componentes del sistema HortiRobot.

La capa de alto nivel está formada por 6 módulos principales: adquisición, detección, decisión, navegación, simulación y control. Estos módulos trabajan de manera jerárquica entre sí (aunque todos los módulos tienen acceso a la información global, ya que al trabajar en ROS la información se publica en *broadcast*), tal y como se observa en la Figura 5. A continuación, se describen con mayor nivel de detalle cada uno de estos módulos.

- Módulo de adquisición. Responsable de extraer la información de color y el mapa de distancias de la escena, mediante la comunicación con los drivers de las cámaras, que están en la capa de bajo nivel. Por tanto, este módulo cuenta con dos nodos ROS. Uno de ellos se encarga de implementar la comunicación con ambas cámaras (RGB y TOF) para adquirir los datos de forma síncrona, y el otro se encarga de registrar los datos de color y de rango en el mismo sistema de referencia (ver Figuras 6 y 7) (Salinas

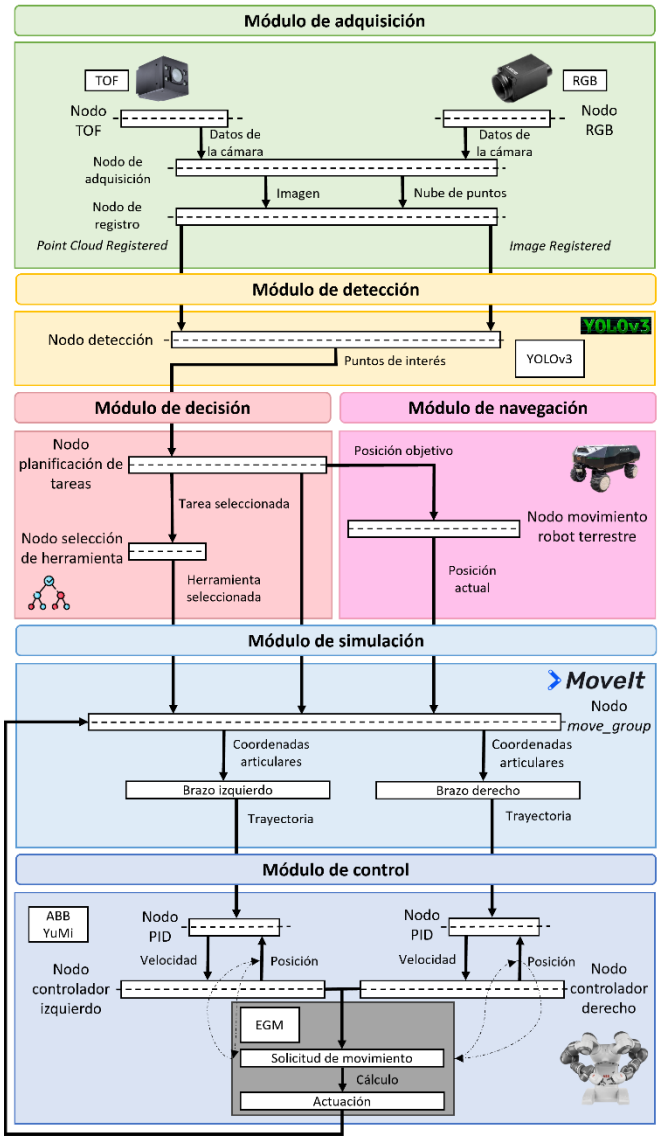

Figura 5: Módulos de alto nivel de HortiRobot.

et al., 2015), (Fernández *et al.*, 2014). De esta forma, en la imagen registrada resultante, se consigue una correspondencia directa entre la información de color y la información espacial proporcionada por la nube de puntos de la cámara TOF.

– Módulo de detección. Incluye algoritmos de detección y de segmentación semántica para la comprensión de la escena, utilizando los datos obtenidos del módulo de adquisición. Estos algoritmos se combinan con otros métodos de estimación para generar, en cada caso, el posicionamiento tridimensional del punto objetivo al cual los efectores finales de los brazos robóticos deben dirigirse. Este punto objetivo puede corresponder a la zona de corte o agarre, dependiendo de la fase de la tarea en curso. El nodo ROS que implementa estos algoritmos incorpora YOLOv3 (*You Only Look Once*), un algoritmo de detección en tiempo real ampliamente utilizado en tareas de localización. YOLOv3 es capaz de predecir directamente las probabilidades de las clases y las correcciones en las coordenadas de la caja delimitadora a partir de imágenes completas utilizando una sola red neuronal convolucional de avance (*feed forward convolutional neural network*). Una característica destacada de YOLOv3 es que elimina la necesidad de generar propuestas de región y realizar remuestreo de características, ya que todas las etapas se encapsulan en una sola red. Esto da lugar a un sistema de detección completo de extremo a extremo que es altamente efectivo en la identificación de objetos y la comprensión del entorno (Redmon y Farhadi, 2018). Por tanto, este módulo es capaz de identificar los objetos de interés de la escena. La capacidad de YOLOv3 de analizar continuamente el entorno permite trabajar de manera dinámica, replanificando la tarea inmediatamente en caso de alteración del entorno, o si se produce una colisión inesperada. Esta red sería común a todas las tareas de HortiRobot, variando sólo la clase de interés en función de los objetivos. A continuación, el nodo ROS aplica un algoritmo de segmentación semántica basado en máquinas de soporte vectorial a las cajas delimitadoras resultantes de la red YOLOv3. De esta forma se seleccionan sólo los píxeles de interés para su procesamiento posterior. A partir de aquí, el nodo aplica diferentes métodos de estimación para generar los puntos de interés en función de cada una de las tareas. En el caso de los frutos, por ejemplo, los píxeles resultantes de la segmentación permiten calcular su centroide. La estimación de los puntos de corte para otras tareas, como el injerto y la poda, a partir de los píxeles resultantes, requiere métodos más complejos. Por tanto, su descripción está fuera del alcance de este artículo.

Figura 6: Imagen adquirida por la cámara RGB.

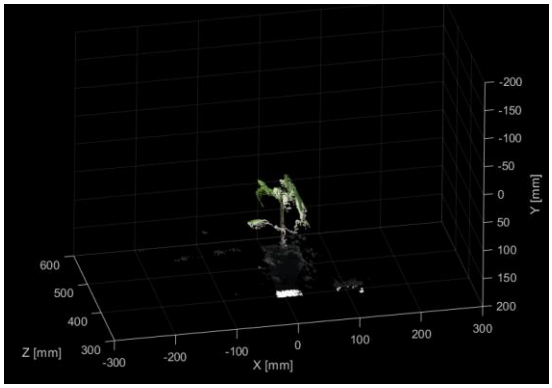

Figura 7: Datos proporcionados por el nodo de registro del módulo de adquisición.

- Módulo de decisión. Incluye los algoritmos que proporcionan al robot la capacidad de decidir de forma autónoma con qué objetivos trabajar durante la tarea, considerando el estado de los brazos manipuladores y las entradas proporcionadas por el sistema de percepción inteligente. Dependiendo de la tarea a ejecutar, el módulo implementará un algoritmo u otro. Por ejemplo, en el caso de recogida de frutos, se han empleado procesos de decisión de Markov Parcialmente Observables (POMDP) para planificar la tarea. Este método permite estimar de manera óptima el valor de la información adquirida y ponderar diversas opciones de acción en un entorno no estructurado. Así, el POMDP podría sugerir la cosecha simultánea de dos frutos con cada uno de los brazos, cosechar un solo fruto con uno de los brazos, o retirar hojas que provocan oclusiones para mejorar la visibilidad de un fruto en particular. Además, puede abordar situaciones complejas, como la dificultad para agarrar un fruto específico debido a su ubicación. El modelo POMDP utiliza el porcentaje de visibilidad del fruto y la proporción de oclusión como parámetros en las probabilidades de observación y agarre de las diferentes partes de la planta. Este módulo también cuenta con algoritmos para seleccionar el efector final adecuado en función de las necesidades de la tarea, así como para definir la orientación con la que deben ejecutar las distintas subtareas. La elección del efector final se comunica al módulo de simulación, de modo que se pueda actualizar su modelo virtual y ser considerado durante la planificación de las trayectorias. Para determinar la orientación del efector final en el caso de la cosecha, se utilizarán los píxeles segmentados en el módulo de percepción para obtener el ángulo sobre la horizontal de cada fruto que se ha de recoger, así como de su tallo, para poder obtener una línea libre de obstáculos en el espacio. Esta línea es la que se utiliza como referencia para la orientación del efector final. En la poda se utiliza un método similar, considerando el ángulo sobre la horizontal de la rama a cortar. En el caso del injerto, la orientación es fija y está definida previamente, al estar restringida por las cuchillas de corte del efector final, y la pinza de sujeción del injerto.

- Módulo de navegación. Integra los algoritmos necesarios para guiar con seguridad al robot móvil terrestre dentro del entorno de aplicación hasta lograr el posicionamiento requerido para la ejecución de la tarea definida. De todas formas, cabe destacar que la navegación en el caso de uso propuesto está bastante limitada de momento. La cosecha, el injerto y la poda se realizan con el RB-VOGUI estático. Una vez que se han cosechado todos los frutos de la escena, se han podado las ramas, o se ha realizado el injerto, es cuando el robot móvil se mueve a su próximo objetivo. Por tanto, una vez finalizada la tarea que se está ejecutando, se envía un comando de desplazamiento horizontal hasta el próximo objetivo. El módulo cuenta con un mapa del entorno y es capaz de detectar obstáculos para evitar colisiones.

- Módulo de simulación. Una vez que el módulo de decisión establece la tarea a realizar y el módulo de detección proporciona los puntos objetivos, se procede al cálculo de las coordenadas articulares de ambos brazos robóticos. Con estas coordenadas se efectúa la generación de las trayectorias de los brazos, que son comunicadas al módulo de control. Para ello, el módulo de simulación cuenta con la presencia de algoritmos de planificación. Estos algoritmos utilizan información sobre los objetos de interés y la configuración articular de los brazos robóticos para modelar previamente el escenario virtual. Esto permite calcular trayectorias libres de colisiones con base en los objetivos establecidos, asegurando que los brazos alcancen las configuraciones necesarias para llevar a cabo las tareas. Para abordar el estudio dinámico de las trayectorias libres de ambos brazos al operar en un mismo espacio de trabajo, además de contar con los algoritmos de planificación de trayectorias, se requiere una descripción detallada de la posición de todos los puntos que forman los brazos robóticos, así como el valor de sus coordenadas articulares en todo momento. Para ello, se define el robot mediante un archivo XML en formato *Unified Robot Description Format* (URDF), y se establece una comunicación continua con cada brazo robótico a través de un tópico de ROS denominado *"/joint_states"*, mediante el cual se reciben en cada momento las coordenadas articulares. De este modo, ambos brazos robóticos tienen un conocimiento preciso y actualizado de la localización de su contraparte en cada instante, permitiendo una coordinación efectiva de su operación conjunta en el mismo entorno de trabajo. En este archivo XML también se incluye el modelo del efector final seleccionado por el módulo de decisión. Los algoritmos mencionados son implementados en nodos ROS, utilizando MoveIt (Coleman *et al*., 2014) y la librería OMPL (*Open Motion Planning Library)* (Ioan *et al*., 2012), (Kingston *et al.*, 2019). El entorno virtual resultante es una reproducción fiel de la escena real, de modo que los diferentes obstáculos detectados son considerados para el cálculo de las trayectorias entre las posiciones finales y deseadas de cada brazo. Este módulo es, por tanto, el encargado de calcular y enviar los comandos correspondientes al módulo de control.

- Módulo de control. Desempeña un papel esencial en el sistema HortiRobot. Su función principal es ajustar las trayectorias de los brazos robóticos utilizando un controlador PID, asegurando así un movimiento similar al simulado en términos de posición y velocidad. Este controlador PID se define en los archivos de configuración de MoveIt específicos para el robot YuMi. Estos archivos detallan los valores de P, I y D necesarios para garantizar un control preciso y efectivo de cada una de las articulaciones del robot. También gestiona la asignación de roles de maestro y esclavo para los brazos según las restricciones de la tarea y se adapta en tiempo real a imprecisiones en el entorno durante la ejecución. Una vez que las trayectorias están calculadas y validadas, el módulo se comunica con los controladores de bajo nivel de los brazos a través de las librerías externas de movimiento EGM (*External Guide Motion*) de ABB, activando las articulaciones según las entradas de control del módulo de simulación. Además, el uso de las librerías

EGM facilita la comunicación continua de la posición real del robot a una frecuencia de 250 Hz mediante el protocolo UDP (*User Datagram Protocol*), permitiendo la simulación en tiempo real y previniendo colisiones potenciales.

Por otra parte, la capa de bajo nivel es fundamental para el control preciso del movimiento de los brazos robóticos manipuladores en el sistema HortiRobot, ya que establece la comunicación directa con el hardware, utilizando para ello las librerías EGM, como se ha mencionado en el módulo de control de la capa de alto nivel. El controlador, gobernado por las librerías EGM, evalúa la viabilidad de los movimientos solicitados en función de la velocidad, aceleración, y límites superior e inferior del valor de las coordenadas articulares, y ejecuta las acciones apropiadas en consecuencia. Una vez que se alcanza el objetivo, el proceso se detiene, y queda a la espera de nuevas instrucciones.

Las librerías EGM se pueden utilizar para transmitir posiciones al controlador del robot en coordenadas cartesianas o articulares, y también son útiles para realizar ajustes en las trayectorias de movimiento previamente programadas. Para habilitar las librerías EGM, es necesario instalar la opción *RobotWare Externally Guided Motion* [689-1] en el controlador del robot. En términos generales, EGM ofrece tres modos de funcionamiento distintos, que se enumeran de forma esquemática para su comprensión:

- *EGM Position Stream.* Envía las posiciones objetivo y las posiciones actuales de las unidades mecánicas (cada uno de los brazos), desde el FlexPendant en código RAPID a un equipo externo.
- *EGM Position Guidance.* Es una interfaz de bajo nivel diseñada para usuarios avanzados que brinda un control altamente receptivo del robot al evitar la planificación de trayectorias. Por tanto, con esta opción, el robot no sigue un camino programado en RAPID previamente, sino la trayectoria generada por el módulo de simulación que se encuentra en el ordenador externo.
- *EGM Path Correction.* La trayectoria del robot es modificada o corregida a partir de mediciones realizadas por un dispositivo externo.

En el caso de uso de HortiRobot, se opta por emplear *EGM Position Guidance,* ya que el objetivo es que el control de bajo nivel del robot esté gobernado completamente por los módulos de la capa de alto nivel implementados en ROS. Por tanto, *EGM Position Guidance* utiliza las posiciones generadas por los módulos de simulación y control de la capa de alto nivel para ejecutar el movimiento de cada uno de los brazos robóticos.

La interfaz *EGM Position Guidance* permite la lectura y escritura de posiciones en el sistema de movimiento a una alta velocidad, con un intervalo de tan solo 4 ms y un retraso de control que oscila entre 10 y 20 ms. También es posible especificar puntos de referencia utilizando valores articulares o una pose, siendo posible definir esta última en cualquier objeto de trabajo que permanezca estático durante el movimiento ejecutado con *EGM Position Guidance*.

Para garantizar un control efectivo con EGM, es esencial tener claros varios aspectos: primero, se debe conocer tanto el punto inicial como el punto final de la trayectoria a seguir. Además, es importante tener en cuenta que el primer movimiento del robot después de reiniciarse no puede ser realizado mediante EGM. Además, EGM no permite movimientos lineales debido a la falta de la funcionalidad de interpolación. No obstante, este problema se aborda mediante el empleo de otros métodos de programación que permiten al robot YuMi emular movimientos lineales, como la planificación en MoveIt. Esta técnica implica almacenar la mayor cantidad posible de puntos de la trayectoria planificada y luego enviar el objetivo al controlador para su ejecución. Aunque este movimiento no será completamente lineal, se aproximará bastante. Además, en situaciones de singularidad, debe tenerse en cuenta que la única solución es desplazarse a otro punto mediante el *jogging*. Estas situaciones pueden surgir debido a errores en la calibración o a una configuración inicial incorrecta de los parámetros del robot. Por lo tanto, en estos casos, el usuario debe realizar manualmente el mantenimiento y la revisión del brazo robótico para reestablecer el sistema al punto de trabajo y llevar a cabo una nueva calibración si es necesario. Por último, si se produce una colisión, se debe reiniciar EGM para evitar problemas adicionales. Los servicios de ROS pueden utilizarse para enviar un mensaje indicando que se ha producido una colisión. En este escenario, es posible continuar el movimiento o realizar uno nuevo desde el punto de colisión, evitando así el error anterior mediante una nueva planificación de la trayectoria.

Por otra parte, para realizar el control del robot, es necesario emplear las librerías RWS (*Robot Web Services*) de ABB en conjunto con las librerías EGM (ver Figura 8).

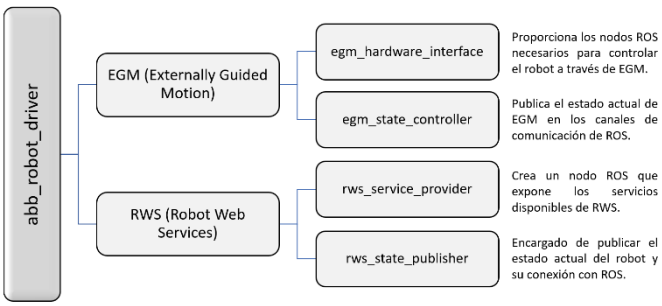

Figura 8: Librerías más relevantes del sistema y su función.

Las librerías RWS proporcionan una serie de servicios y recursos, como el acceso a archivos y directorios, el manejo de la transferencia, creación, eliminación y cambio de nombre de directorios y archivos (similar a FTP), la gestión de subscripciones de recursos y envío de eventos mediante *WebSockets*, el registro de clientes conectados, el manejo de los servicios de RobotWare, como IO, RAPID, E-log y CFG, y el manejo de la funcionalidad global del controlador del robot, como el acceso al reloj del controlador, la identificación del controlador y el reinicio del mismo. En HortiRobot también se utilizan estas librerías para gestionar la asignación de roles entre los brazos robóticos.

Para facilitar la comunicación entre los controladores del robot YuMi de ABB y los módulos de simulación y control de la capa de alto nivel implementados en ROS, se emplean los repositorios *abb_robot_driver* y *abb_libegm* de *ROS_Industrial.* Este código ha sido adaptado a la versión de

ROS con la que se desarrolla la propuesta HortiRobot, realizándose, por tanto, la migración desde ROS Melodic a ROS Noetic. El repositorio *abb_robot_driver* incluye tres paquetes principales: (i) *abb_rws_state_publisher*, que proporciona un nodo ROS que sondea continuamente el controlador del robot YuMi de ABB para detectar estados del sistema, que luego se analizan en mensajes ROS y se publican en el sistema ROS; (ii) *abb_rws_service_provider*, que proporciona un nodo ROS que permite una interacción discreta con el controlador del robot por medio de servicios, como iniciar o detener el programa RAPID y leer o escribir señales de entrada/salida; y (iii) *abb_egm_hardware_interface*, que cuenta con un nodo ROS para ejecutar la interfaz hardware que habilita el control directo del movimiento del robot a través de EGM, y un segundo nodo ROS para detener automáticamente los controladores cuando finalizan las sesiones de comunicación de EGM.

Por otra parte, *abb_libegm* es una librería necesaria para la interacción con los controladores de los robots ABB desde un dispositivo externo. Esta librería trabaja de manera conjunta con *abb_librws*, encargada de los RWS. EGM se comunica a una frecuencia de 250 Hz mediante *Google Protobuf* UDP con el robot, que tiene instalado el sistema al completo gracias a los *Add-in* de *StateMachine y RobotWare*. Por tanto, EGM depende de *Boost C++* y *Google Protocol Bufffers* para poder operar correctamente.

Por otro lado, la librería de RWS llamada *abb_librws* se encarga de que el controlador soporte los *"Robot Web Services"*. Se trata de una librería en C++ que le permite escribir o leer señales de entrada o salida, código en RAPID, e incluso comprobar o modificar el estado del controlador. RWS se comunica vía HTTP y *WebSocket* con TCP (*Transmission Control Protocol).*

En la Figura 9 se indica la conexión del sistema robótico con el controlador y el ordenador externo de forma esquemática. Por tanto, los tres componentes principales de este entorno (ordenador externo, controlador IRC5 y robot YuMi) se comunican entre sí a través de RWS, y el control de movimiento del robot se realiza gracias a las librerías EGM de

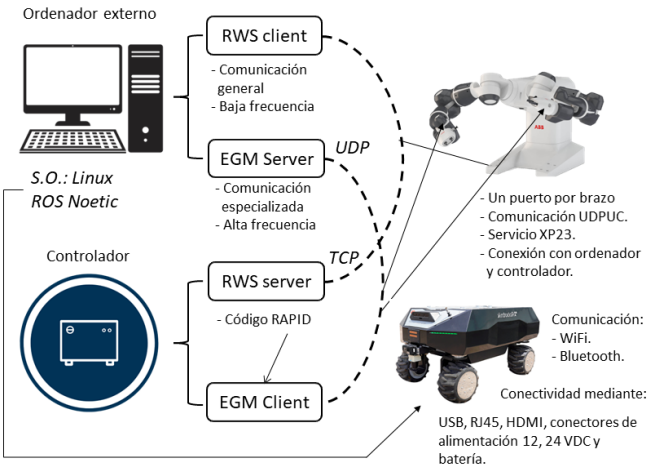

Figura 9: Esquema de conexiones del robot.

movimiento externo de ABB. El controlador es el encargado de crear un servidor de RWS, comunicándose por ethernet con el ordenador externo, que se encarga de comunicarse mediante TCP con el robot, permitiendo de esta forma la obtención de las características de este, así como la ejecución de algunas tareas preprogramadas en RAPID o el control de los SmartGripper. Por otro lado, este ordenador externo envía los comandos deseados de movimiento, calculados en los nodos ROS que componen el módulo de simulación de la capa de alto nivel mediante UDP, a una frecuencia de 250 Hz, al controlador del robot, que actúa como cliente de esta solicitud de EGM y la redirecciona mediante UDP a cada uno de los brazos robóticos, conectados con el ordenador. Cada unidad mecánica del robot dual, es decir, cada uno de sus brazos, se asocia a un puerto determinado del ordenador. Gracias a esta arquitectura, se pueden realizar las tareas para las que fue concebido el sistema HortiRobot.

En la Figura 10 puede observarse una visualización gráfica obtenida mediante el *rqt_graph* de ROS de los nodos y *topics* que componen el sistema. El proceso sigue los pasos explicados en el pseudocódigo de la Figura 11, que muestra la

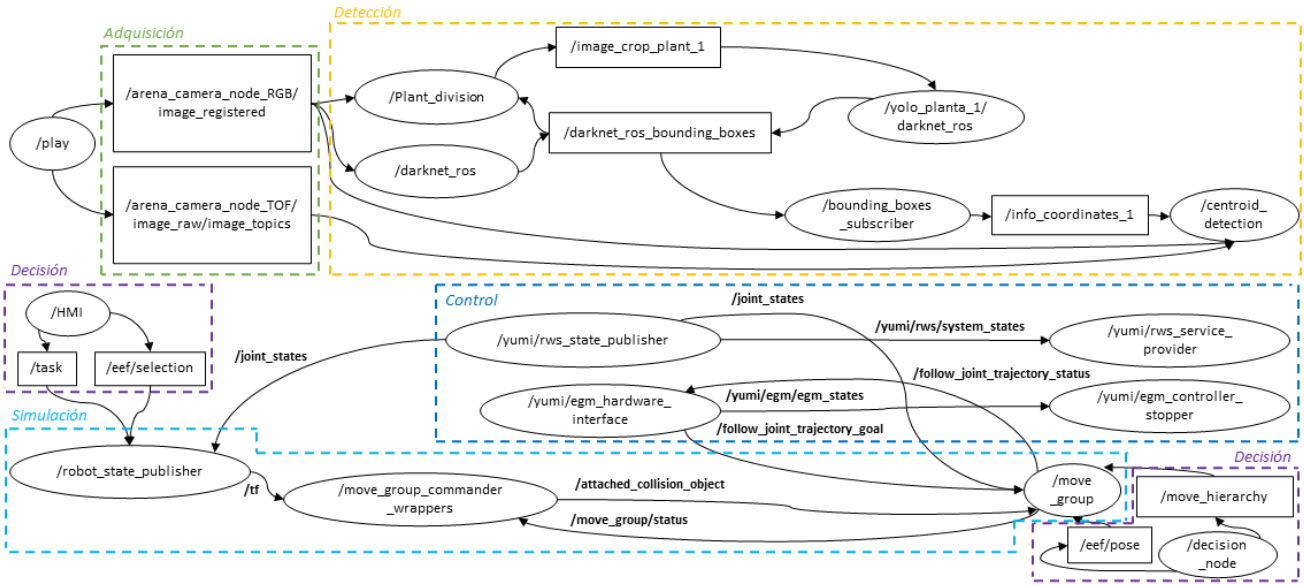

Figura 10: Visualización gráfica de los principales nodos de HortiRobot.

arquitectura general de ROS para la implementación de las tareas realizadas por el sistema robótico HortiRobot. Esta arquitectura es global y puede intercambiarse entre las distintas tareas de este sistema. El nodo */centroid_detection* es input del nodo */move_group*, encargado del movimiento a un punto objetivo del robot. Esta interconexión no se muestra representada en el gráfico para facilitar la comprensión visual del mismo. En la imagen se puede apreciar la relación entre los nodos ROS y los distintos módulos presentes en la arquitectura propuesta.

Algorithm 1 HortiRobot Pseudocode

```
procedure DEFINED TASK
    if detection is required then
        while "plant" not in "image_rgb" do ▷ Using yolo and darknet_ros
            collect image_data
        objective ← point_in_plant ▷ Obtained from TOF camera
    else
        objective ← /follow_joint_trajectory/goal
    publish objective
    subscribe to objective
    egm True
    while egm True do
        while /joint_states not in objective do
            read /joint_states
            read controller
            move robot joints via EGM controller
        gripper_command ▷ Grasp or cut
        egm False
```

Figura 11: Esquema general de la implementación de tareas en HortiRobot.

A nivel de interfaz, el usuario podrá seleccionar la tarea a ejecutar, configurar y monitorizar el sistema robótico, ajustar los parámetros del sistema de percepción y acceder de forma remota a las cámaras que lo conforman. La interfaz diseñada también permite al usuario ejecutar el código de manera inmediata, ofreciendo una comprensión clara de la etapa en la que se encuentra el proceso en todo momento. Además, proporciona información sobre posibles errores en caso de eventos inesperados, así como el estado actual de la ejecución, mostrando el porcentaje completado. Esto resulta especialmente útil para depurar los diferentes módulos que componen el sistema (ver Figura 12). Para la selección de la tarea, el usuario dispone de un menú en el que se muestran todas las opciones disponibles: injerto, poda y cosecha. La tarea seleccionada se muestra también en la interfaz gráfica. Además, como cada una de estas tareas se desglosa en distintas subtareas, la interfaz gráfica también permite visualizar la subtarea que se está ejecutando bajo el título "objetivo actual". Por otra parte, el estado de las cámaras y del robot, se muestra de forma binaria, indicando si las cámaras están conectadas o desconectadas, y si el robot está en movimiento o detenido. Por último, se presenta información sobre el estado de apertura de los efectores finales.

4. Conclusiones y líneas futuras

A día de hoy no existe una arquitectura de software de referencia que sea la más adecuada para una aplicación robótica determinada. Sin embargo, su diseño es fundamental para lograr una integración funcional de los diversos procesos requeridos. En este artículo se presenta el diseño de una arquitectura software basada en capas, concebida para facilitar la implementación, prueba, depuración y evaluación de nuevos algoritmos de percepción, toma de decisiones, manipulación y control en entornos no estructurados para robots bimanipuladores móviles autónomos. El concepto basado en capas permite incrementar la flexibilidad y la capacidad de operar en múltiples niveles de abstracción simultáneamente, así como mejorar la adaptabilidad a entornos cambiantes y la gestión de recursos. Para demostrar la viabilidad de la propuesta, la implementación de la arquitectura se valida mediante su aplicación al sistema robótico de manipulación dual HortiRobot, concebido para realizar varias de las tareas implicadas en el ciclo de vida de los cultivos agrícolas. En este

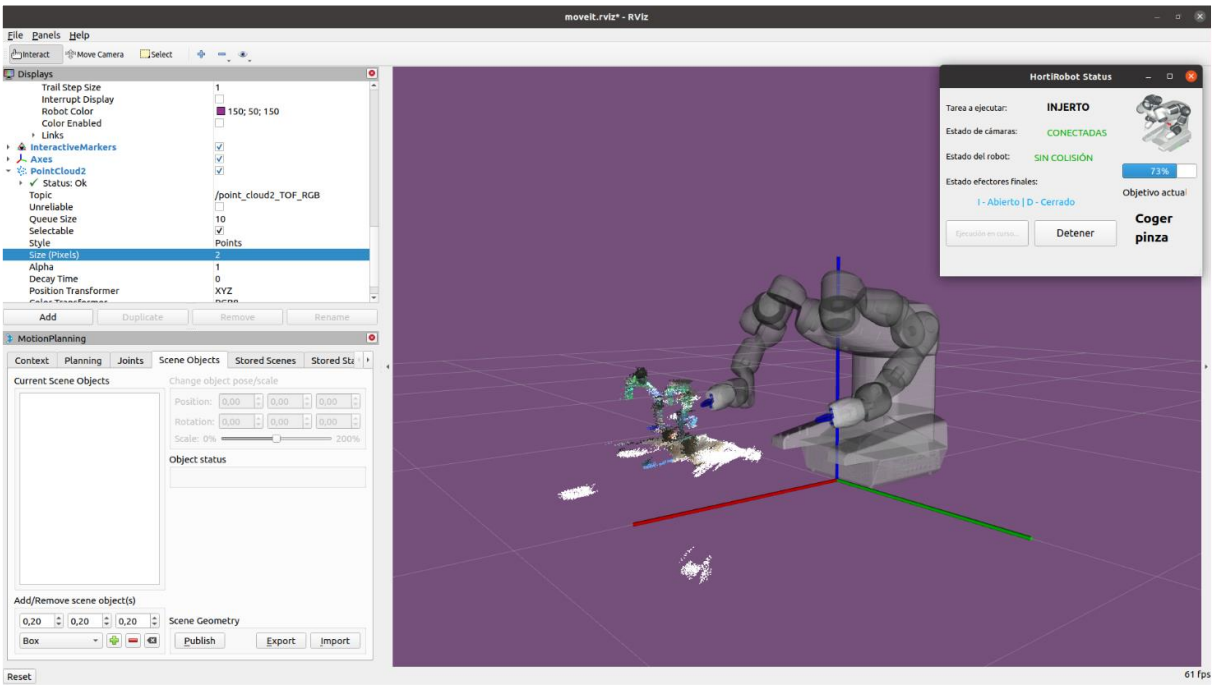

Figura 12: Visualización de la interfaz de usuario de HortiRobot.

caso de uso propuesto, la arquitectura diseñada controla, habilita la comunicación e integra eficazmente los distintos módulos que proporciona al sistema robótico la inteligencia necesaria para percibir el entorno y responder eficazmente a los objetivos de las tareas planteadas, minimizando además el riesgo inherente de daño a los equipos físicos, gracias al entorno de simulación programado. La arquitectura software implementada para el sistema HortiRobot se compone de tres capas principales: una capa de alto nivel, una capa de bajo nivel y otra capa está destinada a la interfaz de usuario.

Los seis módulos que constituyen la capa de alto nivel gestionan la interpretación de la información adquirida por el sistema de percepción, la toma de decisiones para llevar a cabo la tarea definida por el usuario, la planificación de las trayectorias libres de colisiones, la monitorización de la ejecución de cada tarea hortofrutícola, abordando las subtareas que la componen, y la gestión de contingencias inesperadas. Simultáneamente, la capa de bajo nivel resulta fundamental para el control preciso de los movimientos de los brazos robóticos manipuladores del sistema HortiRobot. El uso de las librerías EGM de ABB en esta capa, ha demostrado ser un componente esencial en el proceso, ya que establecen la comunicación directa con el hardware, permitiendo un control efectivo del robot.

Entre los trabajos futuros previstos, cabe mencionar, el diseño de algoritmos de aprendizaje para la transferencia de habilidades o destrezas, que combinarán la toma de decisiones con el aprendizaje por demostración para mejorar la adaptabilidad del sistema robótico a los diferentes escenarios complejos y a la variabilidad de productos y herramientas.

Agradecimientos

Esta publicación es parte del proyecto de I+D+i HortiRobot (Comprensión visual de la escena y aprendizaje inteligente para la manipulación dual robótica en tareas hortícolas) / ayuda PID2020-116270RB-I00, financiado por MCIN/AEI/ 10.13039/501100011033 y del proyecto DigiHortiRobot (Gemelo Digital para la Robótica Hortícola Sostenible del Futuro en Invernaderos) / ayuda TED2021-132710B-I00, financiado por MCIN/AEI/10.13039/501100011033 y por la Unión Europea NextGenerationEU/PRTR.

Referencias

Bouguet, J.-Y., 2022. Camera Calibration Toolbox for Matlab. CaltechDATA. DOI: 10.22002/D1.20164

Cervantes, H., Velasco-Elizondo, P., Castro, L., 2016. Arquitectura de Software. Conceptos y ciclo de desarrollo. CENGAGE Learning

Chiabrando, F., Chiabrando, R., Piatti, D., Rinaudo, F., 2009. Sensors for 3D Imaging: Metric Evaluation and Calibration of a CCD/CMOS Time-of-Flight Camera. Sensors, 9(12), 10080. DOI: 10.3390/s91210080

Chikurtev, D., 2022. Service-oriented architecture for control of modular robots. International Conference on Circuits, Systems, Communications and Computers, 304-309. DOI: 10.1109/CSCC55931.2022.00059

Coleman, D., Sucan, I.A., Chitta, S., Correll, N., 2014. Reducting the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study. Journal of Software Engineering for Robotics, 5(1): 3-16. DOI: 10.6092/JOSER_2014_05_01_p3

Duan, J., Gong, X., Zhang, Q., 2023. A digital twin-driven monitoring framework for dual-robot collaborative manipulation. Int J Adv Manuf Technol 125, 4579-4599. DOI: 10.1007/s00170-023-11064-2

Eurostat, 2021. Economic accounts for agriculture – values at current prices. https://ec.europa.eu/eurostat/databrowser/view/aact_eaa01/default/table?lang=en, 2016-2021

Fernández, R., Salinas, C., Montes, H., Sarria, J., 2014. Multisensory System for Fruit Harvesting Robots. Experimental Testing in Natural Scenarios and with Different Kinds of Crops. Sensors, 14, 23885-23904. DOI: 10.3390/s141223885

Ferrati, M., Settimi, A., Muratore, L., Cardellino, A., Rocchi, A., Mingo, H.E., Pavan, C., Kanoulas, D., Tsagarakis, N.G., Natale, L., Pallottino, L., 2016. The Walk-Man Robot Software Architecture. Front. Robot. AI, 3. DOI: 10.3389/frobt.2016.00025

Geer, L., Gu, D., Wang, F., Mohan, V., Dowling, R., 2022. Novel Software Architecture for an Autonomous Agricultural Robotic Fruit Harvesting System. 27th international Conference on Automation and Computing, 1-6. DOI: 10.1109/ICAC55051.2022.9911161

Gillini, G., Di Lillo, P., Arrichiello, F., Di Vito, D., Marino, A., Antonelli, G., Chiaverini, S., 2022. Industrial Robot: the international journal of robotics research and application. 49(1), 11-20. DOI: 10.1108/IR-07-2020-0137

Gribov, V., Voos, H., 2014. A multilayer software architecture for safe autonomous robots. Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA), 1-8. DOI: 10.1109/ETFA.2014.7005217

Hendrich, N., Bistry, H., Zhang, J., 2015. Architecture and Software Design for a Service Robot in an Elderly-Care Scenario. Engineering, 1(1), 027-035. DOI: 10.15302/J-ENG-2015007

Ibarguren, A., Eimontaite, I., Outón, J.L., Fletcher, S., 2020. Dual Arm Co-Manipulation Architecture with Enhanced Human–Robot Communication for Large Part Manipulation. Sensors., 20(21), 6151. DOI: 10.3390/s20216151

Ioan, A., Moll, M., Kavraki, L., 2012. The Open Motion Planning Library, IEEE Robotics & Automation Magazine, 19(4), 72–82. https://ompl.kavrakilab.org

Kalaycioglu, S., De Ruiter, A. 2023. Dual arm coordination of redundant space manipulators mounted on a spacecraft. Robotica, 41(8), 2489-2518. DOI:10.1017/S0263574723000504

Kingston, Z., Moll, M., Kavraki, L. E., 2019. Exploring Implicit Spaces for Constrained Sampling-Based Planning, International Journal of Robotics Research, 38(10–11), 1151–1178. DOI: 10.1177/0278364919868530

Kortenkamp, D., Simmons, R., Brugali, D., 2016. Robotic Systems Architectures and Programming, Springer Handbook of Robotics. Springer Handbooks, 283-306. DOI: 10.1007/978-3-319-32552-1_12

Medvidovic, N, Tajalli, H., Garcia, J., Krka, I., Brun, Y., Edwars, G., 2011. Engineering Heterogeneous Robotics Systems: A Software Architecture-Based Approach. Computer., 44(5), 62-71. DOI: 10.1109/MC.2010.368

Ministerio de Agricultura, Pesca y Alimentación. Gobierno de España, 2021. Producciones agrícolas. Cifras del Sector de Frutas y Hortalizas. https://www.mapa.gob.es/es/agricultura/temas/producciones-agricolas/frutas-y-hortalizas/informacion_general.aspx

Nakagawa, E. Y., Antonino, P. O., Schnicke, F., Capilla, R., Kuhn, T., Liggesmeyer, P., 2021. Industry 4.0 reference architectures: State of the art and future trends. Computers & Industrial Engineering, 1-13. DOI: 10.1016/j.cie.2021.107241

Phanomchoeng, G., Saadi, M., Sasithong, P., Tangmongkhonsuk, J., Wijayasekara, S.K., Wuttisittikulkij, L., 2020. Hardware software co-design of a farming robot. Engineering Journal. 24(1), 1-10. DOI: 10.4186/ej.2020.24.1.199

Pramod U., C., Murugan, M., Prajakta, C., 2015. A review on Software Architecture Styles with Layered Robotic Software Architecture. International Conference on Computing Communication Control and Automation. 827-831. DOI: 10.1109/ICCUBEA.2015.165

Richards, M, 2015. Software Architecture Patterns. O'Reilly Media Inc.

Sadjadi, E.N., Fernández, R. Challenges and Opportunities of Agriculture Digitalization in Spain, 2023. Agronomy, 13, 259. DOI:10.3390/agronomy13010259.

Salinas, C., Fernández, R., Montes, H., Armada, M. 2015. A New Approach for Combining Time-of-Flight and RGB Cameras Based on Depth-Dependent Planar Projective Transformations. Sensors, 15: 24615-24643. DOI: 10.3390/s150924615

Vrochidou, E., Tziridis, K., Nikolau, A., Kalampokas, T., Papakostas, G., Pachidis, T. P., Mamalis, S., Koundouras, S., Kaburlasos, V. G., 2021. An Autonomous Grape-Harvester Robot: Integrated System Architecture. Electronics. 10(9), 1056. DOI: 10.3390/electronics10091056

Xu, R., Li, C., 2022. A modular agricultural robotic system (MARS) for precision farming: Concept and implementation. Journal of Field Robotics. 39(4), 387-409. DOI: 10.1002/rob.22056